(19)日本国特許庁（ＪＰ）　(12) **公開特許公報（A）**　(11)特許出願公開番号

特開平7－225770

(43)公開日　平成7年(1995)8月22日

(51)Int.Cl.[6]	識別記号	庁内整理番号	ＦＩ		技術表示箇所
Ｇ０６Ｆ　17/30					
		9194－5L	Ｇ０６Ｆ　15/ 419	３１０	

審査請求　未請求　請求項の数３　ＦＤ　（全 15 頁）

(21)出願番号　特願平6－36346

(22)出願日　平成６年(1994)２月10日

(71)出願人　000005496
富士ゼロックス株式会社
東京都港区赤坂三丁目３番５号

(72)発明者　沼田　賢一
神奈川県横浜市保土ヶ谷区神戸町134番地
横浜ビジネスパークイーストタワー　富士ゼロックス株式会社内

(74)代理人　弁理士　南野　貞男　（外３名）

(54)【発明の名称】　データ検索装置

(57)【要約】

【目的】　ノード間の関係が木あるいは木の集合で表現されるノードの集合から、第１の条件を満たすノードに対して祖孫関係を持ちかつ第２の条件を満たすノードを検索するデータ検索装置を提供する。

【構成】　入力手段が第１の条件，第２の条件およびその間の祖孫関係を入力する。ノード検索手段が第１の条件および第２の条件を受け取り、索引保持手段の索引から第１の条件を満たすノードに対応するレコードの集合と、第２の条件を満たすノードに対応するレコードの集合とを抽出する。接続関係検索手段は、祖孫関係を受け取り、前記ノード検索手段の抽出した一方のレコードの指す部分木索引の中に、他方のレコードの指すノードが存在するか否かを検索し、該当するノードが存在し、かつ前記祖孫関係を満足する場合に、該当ノードを第２の条件を満たすノードとして出力する。

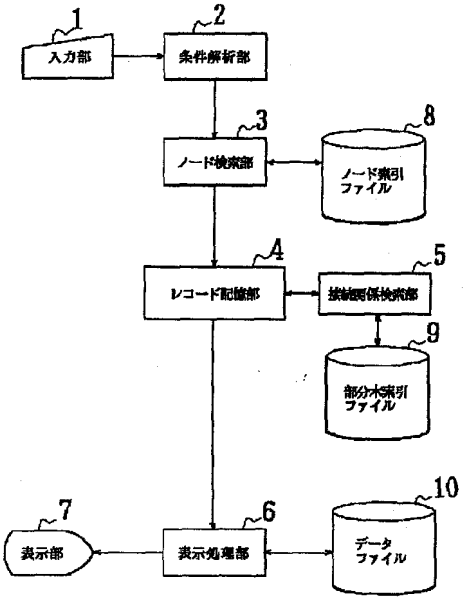

図１　データ検索装置の構成

1

【特許請求の範囲】

【請求項1】　ノード間の関係が木あるいは木の集合で表現されるノードの集合から、第1の条件を満たすノードに対して祖孫関係を持ちかつ第2の条件を満たすノードを検索するデータ検索装置であって、

第1の条件，第2の条件およびその間の祖孫関係を入力する入力手段と、

ノード，ノード間の関係およびノードの属性値を格納したデータ記憶手段と、

前記データ記憶手段に格納された葉ノード以外の各ノードごとに設けられ、当該ノードを根ノードとする部分木を構成する各ノードへのポインタから成る部分木索引を保持する部分木索引記憶手段と、

前記データ記憶手段に格納された各ノードごとに設けられ、当該ノードへのポインタと前記部分木索引へのポインタとから成るレコードの集合を保持するレコード集合保持手段と、

前記データ記憶手段に格納されたノードの属性値と該属性値を持つノードに対応する前記レコードへのポインタから成る索引を保持する索引保持手段と、

前記入力手段から入力された第1の条件および第2の条件を受け取り、前記索引保持手段の索引から第1の条件を満たすノードに対応するレコードの集合と、第2の条件を満たすノードに対応するレコードの集合とを抽出するノード検索手段と、

前記入力手段から入力された第1の条件と第2の条件の間の祖孫関係を受け取り、前記ノード検索手段の抽出した一方のレコードの指す部分木索引の中に、他方のレコードの指すノードが存在するか否かを検索し、該当ノードが存在し、かつ前記祖孫関係を満足する場合に、該当ノードを第2の条件を満たすノードとして出力する接続関係検索手段と、

該接続関係検索手段の出力する該当ノードの持つ情報を前記データ記憶手段から抽出して表示する表示処理手段とを備えることを特徴とするデータ検索装置。

【請求項2】　請求項1に記載のデータ検索装置において、

前記部分木索引記憶手段は、更に、前記部分木索引のポインタの指す各ノードが、該部分木索引の根ノードとなるノードの子であるか子以外の子孫であるかを区別するデータを含むことを特徴とするデータ検索装置。

【請求項3】　ノード間の関係が木あるいは木の集合で表現されるノードの集合から、第1の条件を満たすノードに対して祖孫関係を持ちかつ第2の条件を満たすノードを検索するデータ検索装置であって、

第1の条件，第2の条件およびその間の祖孫関係を入力する入力手段と、

ノード，ノード間の関係およびノードの属性値を格納したデータ記憶手段と、

前記データ記憶手段に格納された各ノードごとに設けられ、当該ノードへのノードポインタと当該ノードの親ノードへのノードポインタを保持するレコードを指す親ポインタとから成るレコードの集合を保持するレコード集合保持手段と、

前記データ記憶手段に格納されたノードの属性値と該属性値を持つノードに対応する前記レコードへのポインタから成る索引を保持する索引保持手段と、

前記入力手段から入力された第1の条件および第2の条件を受け取り、前記索引保持手段の索引から第1の条件を満たすノードに対応するレコードの集合と、第2の条件を満たすノードに対応するレコードの集合とを抽出するノード検索手段と、

前記入力手段から入力された第1の条件と第2の条件の間の祖孫関係を受け取り、前記ノード検索手段の抽出した一方のレコードの内、子孫である一方のレコードの親ポインタを根ノードまで辿り、辿った経路上のレコードのノードポインタの集合と、第2の条件を満たすレコードのノードポインタの集合との積集合を計算して得られるノードポインタの指すノードを該当ノードとして出力する接続関係検索手段と、

該接続関係検索手段の出力する該当ノードの持つ情報を前記データ記憶手段から抽出して表示する表示処理手段とを備えることを特徴とするデータ検索装置。

【発明の詳細な説明】

【0001】

【産業上の利用分野】本発明は、データ検索装置に関し、特に、ノード間の関係が木あるいは木の集合で表現されるノードの集合から、ノードの条件とノード間の接続関係の条件を利用して検索を行う場合に、検索の処理を高速化するデータ検索装置に関する。

【0002】

【従来の技術】従来から、データ内容に応じてデータ処理の取扱いを容易とするため、データを構造化し、木構造で表現されるデータ構造として扱うデータ処理装置が開発されている。例えば、文書処理装置で扱う文書データは、文書内容がその概念構造から章，節，項などの階層を有する論理構造で表現されるので、木構造で表現されるデータ構造として扱うことができる。

【0003】このように、木構造で表現されるデータ構造の一例として文書の論理構造がある。文書の論理構造は、図8に示すように、一般に有向順序木により表現することができる。図8は、文書の論理構造を木構造で表現している一例を示す図である。図8に示す文書の論理構造の例では、報告書という文書は、「報告書」という根ノードに結びついて、標題，著者名，第1章の内容，第2章の内容，第3章の内容が、第1階層のノードとなる論理構造で構成されている。そのノードの第1章の内容は「見出し」と3つの「段落」からなる第2階層のノードから構成されている。第2章の内容は「見出し」と「図」と2つの「段落」からなる第2階層のノードから

構成されている。同様に、第3章の内容は「見出し」と「図」と2つの「段落」からなる第2階層のノードから構成されている。このような文書の論理構造となっている。

【0004】木構造で表現された文書の論理構造のデータの検索においては、木構造データの要素の個々のノードの条件のみでなく、ノード間の親子関係あるいは祖孫関係も条件として利用して、条件検索を行うデータ検索処理が行われる。例えば、図8に示すような報告書という文書の論理構造において、「直下の見出しに“データベース”という文字列を含み、かつ、図を含む章を見つけよ」というような条件検索である。

【0005】この場合、「“データベース”という文字列を含む見出し」，「章」，「図」などの条件がノード自体に対する条件であり、「直下」と「含む」がそれぞれ親子関係と祖孫関係などのノード間の関係を規定する条件となる。したがって、ここでは「“データベース”という文字列を含む見出し」，「章」，「図」などのノード自体に対する条件を満たすノードは、それぞれ複数あるが、親子関係あるいは祖孫関係のノード間の関係の条件を満たすノードは、第2章のノードだけとなる。このように、ノード自体に対する条件に、更にノード間の関係の条件までを含めて条件検索を行うことによって、データ検索では、より絞り込まれた回答が得られる。図8に示す例では、「第2章」だけが条件を満たすノードとして検索される。

【0006】また、上記の条件検索の例では、「“データベース”という文字列を含む見出しを直下に持つ章」の第1の集合と「下位に図を含む章」の第2集合の積集合を取ることによって、その検索結果が回答として得られる。この場合においては、第1の集合を求める検索と、第2の集合を求める検索では、それぞれの条件検索が、個々のノードの条件とノード間の親子関係あるいは祖孫関係の条件を用いた検索となっている。

【0007】一般に、木構造で表現されるデータ構造のデータにおいて、個々のノードの条件のみでなく、ノード間の親子関係あるいは祖孫関係の条件も用いた検索を行う検索処理では、第1の条件を満たすノードと親子関係あるいは祖孫関係を持ち、かつ、第2の条件を満たすノードを検索するというデータ検索になる。

【0008】そこで、木構造で表現されるデータ構造の各々のデータに対して、第1の条件を満たすノードと親子関係あるいは祖孫関係を持ち、かつ第2の条件を満たすノードを検索する検索処理を高速に行う場合には、次のような検索処理方法を用いることになる。

【0009】すなわち、データ構造として木構造が用いられる場合、一般的には、データの要素の各ノードをレコード型のデータで表し、各データの間のリンク状態をポインタで表すようにデータ形式を定めている。このため、上記のデータ検索処理の方法は、最も簡易な処理手順によると、木構造のデータ構造から定められる順番（先行順や幅優先順など）に従って、各レコードのリンク状態を示すポインタを順次に辿っていき、各レコード（ノード）が第1の条件を満たすかどうかを調べて、もし、第1の条件を満たしていれば、そのレコードを起点として、ポインタを辿って親子関係あるいは祖孫関係にあるレコードを探し、第2の条件を満たすかどうかを調べる方法が利用される。

【0010】このような検索処理の方法を高速化する場合、第1の条件を満たすノードの検索を行う第1の処理と、第1の条件を満たすノードを起点として、そのノードと親子関係あるいは祖孫関係を持ち、第2の条件を満たすノードの第2の処理との2つの処理に分けて、各々の検索を高速化することになる。

【0011】したがって、このようなデータ検索処理の方法では、例えば、第1の処理においては、逆参照が可能となる転置ファイルを設けることにより、木構造の中の全ノードに対して、ノードの属性値から該当するノードを検索し、これにより、第1の条件を満たすノードの検索を高速に処理する。

【0012】また、第2の処理を高速化するには、例えば「Chris Clifton and Hector Garcia-Molina, “Indexing in a Hypertext Database”, Proceedings of 16th International Conference on VLDB, pp.36-49, 1990」に論じられているような検索方法が利用できる。この方法は、木構造データの葉ノードを除く全てのノードに対して、そのノードを根とする部分木のための転置ファイルを設けるという方法である。

【0013】図9は、ノードの親子関係あるいは祖孫関係の条件を用いる検索処理を、部分木に対する転置ファイルを用いて行う場合の検索方法を説明する図である。この場合の検索方法では、図9に示すように、木構造を構成するノードの各ノードデータに対し、根ノード90，Aノード91，Bノード92のそれぞれに転置ファイル90a，91a，92aを設ける。ここで設ける転置ファイル90a，91a，92aにおいては、各ノードの属性値からの逆参照が可能なように、ノードの属性値とその属性値を持つノードへのポインタ、およびそのノードと転置ファイルを設けたノード（部分木の根）との接続関係（親子関係あるいは祖孫関係）の対応関係のデータが保持される。

【0014】例えば、Aノード91に設ける転置ファイル91aにおいては、当該Aノード91を根とする部分木の各ノードの属性値（bird, dog, fish)と、この属性値を持つノードへのポインタ情報（Cノード，Fノード，Eノード）と、その接続関係の情報（子，子孫の関係）の対応関係との各データを保持している。この接続関係の情報は、親子関係（c）と祖孫関係（d）を区別するために用いられる。

【0015】図9に示すような部分木に対する転置ファ

5

イルを用いて検索処理を行う方法ににより、上記の検索処理における第２の処理を高速化できる。すなわち、第１の条件を満たすノードを起点とし、その子あるいは子孫の関係にある第２の条件を満たすノードの検索処理が、起点となるノードに対応して設けられた転置ファイルを用いて、そこに保持しているノードの属性値を検索することにより、第２の条件を満たすノードを容易にしかも高速に検索することができる。これにより、子あるいは子孫ノードを順次に全て探索するよりも、はるかに高速に処理を行うことができる。

【００１６】一方、上記の検索処理における第２の処理に関連して、第１の条件を満たすノードを起点とし、その親あるいは先祖である第２の条件を満たすノードの検索処理を行う場合、例えば、「第２の条件を満たすノードであり、かつ子あるいは子孫に第１の条件を満たすノードを持つもの」というように、第２の条件を満たすノードを先に検索するようにすれば、図９に示す検索処理方法でも、同様に検索が可能である。

【００１７】また、第１の条件を満たすノードを起点とし、その親あるいは先祖である第２の条件を満たすノードの検索処理は、図９に示すように、各部分木に対して転置ファイルを設ける方法の他に、例えば、図１０に示すように、各ノードに自分の親ノードへのポインタを持たせることにより行う方法がある。このため、木構造を構成する場合のデータ構造では、双方向リンクで関係づけられた構造とする。

【００１８】図１０は、一つのノードからその先祖ノードを辿るために必要なパスを有するデータ構造の一例を示す図である。図１０に示すように、このデータ構造では、木構造を構成する場合に、双方向リンクで接続されており、あるノードを起点とし、その親あるいは先祖であるノードを辿ることもできる。また、この場合、各々のノードにおける情報が２次記憶上でどのように割り付けられているかによって、更に２つの態様に分けることができる。そこで、まず、ノード情報を２次記憶上へ割り付ける場合の２つの異なる態様を説明する。

【００１９】第１の態様は、図１１に示すように、１つのノードにかかる情報を２次記憶上の連続した領域に割り付ける態様であり、また、第２の態様は、図１２に示すように、一つのノードの情報を２次記憶上で複数の領域に分割して割り付ける態様である。

【００２０】第１の態様によるデータ構造では、１つのノードにかかる情報は、２次記憶上の連続した領域に割り付けられる。つまり、図１１に示すように、１つのノードの情報１１０は、親ノードへのポインタ１１１と、複数の子ノードへのポインタ１１２と、m個の属性１１３から構成される。これらは２次記憶上の連続する領域に割り付けられる。属性のフィールドにはそれぞれのノードで保持される実際の値が格納される。

【００２１】また、第２の態様によるデータ構造では、

6

１つのノードの情報を２次記憶上で複数の領域に分割して割り付ける態様である。この場合、１つのノードにかかる一群の情報の分割の仕方は様々であるが、例えば、図１２に示すように、ノード間のリンク情報とノードの内容の情報とに分離することによって、木構造中のリンクを辿る操作は、リンク情報のみを２次記憶から読み出して処理できるので、アクセス処理を高速に処理できることになる。すなわち、図１２に示すようなデータ構造においては、１つのノードのノード情報はノード間のリンク情報１２１のみから構成される。リンク情報１２１は、親ノードへのポインタ１２１ａと、子ノードの集合へのポインタ１２１ｂと、属性集合へのポインタ１２１ｃを持っている。ここでのポインタ１２１ｂによって、指示される子ノードの集合１２２と、ポインタ１２１ｃにより指示される属性の集合１２３は、ノード間のリンク情報１２１とは分割されて別の２次記憶の領域に割り付けられる。

【００２２】次に、このような２つの態様で構成されたデータ構造のノードのそれぞれについて、あるノードから、与えられた条件を満たすその親あるいは先祖ノードを検索する場合の処理について説明する。その処理は次のように行われる。

【００２３】第１の態様のデータ構造（図１１）では、あるノードの情報１１０を２次記憶から主記憶に読み込み、そのノードの中の１つのフィールドである親ノードへのポインタ１１１を調べ、次に、その親ノードの情報を２次記憶から主記憶に読み込む。そして、読み込んだノードの属性１１３の部分の情報を調べて、与えられた条件を満たしているかどうかを調べる。先祖ノードを検索する場合は、このような操作を順次に繰り返すことになる。

【００２４】第２の態様のデータ構造（図１２）では、同じく、あるノード情報１２０を２次記憶から主記憶に読み込むが、この場合には、ノード間のリンク情報１２１のみを読み込む。そして、そのノード間のリンク情報１２１の中の１つのフィールドである親ノードへのポインタ１２１ａを調べ、親ノードの情報を２次記憶から主記憶に読み込む。そして、読み込んだ親ノードのノード間のリンク情報の１つのフィールドである属性集合へのポインタ１２１ｃを調べ、対応する属性集合を２次記憶から主記憶に読み込み、次に、読み込んだ属性集合の中の属性の情報を調べて、与えられた条件を満たすかどうか調べる。先祖ノードを検索する場合は、上記の操作を順次に繰り返す。

【００２５】

【発明が解決しようとする課題】ところで、図９に示すように、葉ノードを除く全てのノードに対して、そのノードを根とする部分木のための転置ファイルを設ける方法では、各転置ファイルにノードの重複があり、転置ファイルの合計サイズが増大し、必要なメモリ量が増大す

るという問題点がある。例えば、木をノード数nの平衡木とすると、転置ファイルのレコード数は、爆発的（nlogn）に増大する。

【0026】また、各ノードに自分の親ノードへのポインタを持たせ、そのポインタをもとに親あるいは先祖ノードへアクセスして、条件を満たすかどうかを調べるという方法では、順次に各々のノードをアクセスして検索して行くので、最終的に条件に適合するノードに辿りつくまでに無駄なノードのアクセスがあり、検索時間コストが大きいという問題がある。また、1つのノードが、図11に示すように、第1の態様で2次記憶上に割り付けられている場合、特に、1つのノードの情報のサイズが大きくなると、ノードを2次記憶から主記憶へ読み込むのに時間がかかり、更に検索時間コストが大きくなるという問題がある。

【0027】一方、一つのノードの情報が、図12に示すように、第2の態様によってノード間のリンク情報が区別されて別に2次記憶上に割り付けられている場合であっても、あるノードの親ノードが与えられた条件を満たすかどうかを調べるためには、あるノードから親ノードへのポインタを辿って親ノードを得る操作と、得られた親ノードから更に属性集合へのポインタを辿ってノードの内容の情報を得る操作とが必要である。したがって、この場合、2次記憶から主記憶へのデータの読み込みを2回行う必要があり、全体として、その検索時間コストがあまり低下しないという問題もある。

【0028】したがって、本発明の目的は、ノード間の関係が木あるいは木の集合で表現されるノードの集合から、第1の条件を満たすノードに対して祖孫関係を持ちかつ第2の条件を満たす第2のノードを高速に検索することができるデータ検索装置を提供することにある。

【0029】また、本発明の第2の目的は、第1の条件を満たすノードを起点として、そのノードと親子関係あるいは祖孫関係を持つ第2の条件を満たすノードの検索を行うための転置ファイルをノードの属性値の情報を除いた部分木索引に置き換え、索引を行う場合の記憶容量を低減し、処理速度を向上させることができるデータ検索装置を提供することにある。

【0030】また、本発明の第3の目的は、第1の条件を満たすノードを起点として、その親あるいは先祖である第2の条件を満たすノードを検索する場合に、2次記憶から主記憶へのデータの読み込み回数を極力減らし、高速に親あるいは先祖ノードへアクセスすることができるようにしたデータ検索装置を提供することにある。

【0031】

【問題点を解決するための手段】上記の目的を達成するため、本発明のデータ検索装置（請求項1）は、ノード間の関係が木あるいは木の集合で表現されるノードの集合から、第1の条件を満たすノードに対して祖孫関係を持ちかつ第2の条件を満たすノードを検索するデータ検索装置であって、第1の条件，第2の条件およびその間の祖孫関係を入力する入力手段（1，2）と、ノード，ノード間の関係およびノードの属性値を格納したデータ記憶手段（10）と、前記データ記憶手段に格納された葉ノード以外の各ノードごとに設けられ、当該ノードを根ノードとする部分木を構成する各ノードへのポインタから成る部分木索引を保持する部分木索引記憶手段（9）と、前記データ記憶手段に格納された各ノードごとに設けられ、当該ノードへのポインタと前記部分木索引へのポインタとから成るレコードの集合を保持するレコード集合保持手段（14）と、前記データ記憶手段に格納されたノードの属性値と該属性値を持つノードに対応する前記レコードへのポインタから成る索引を保持する索引保持手段（12）と、前記入力手段から入力された第1の条件および第2の条件を受け取り、前記索引保持手段の索引から第1の条件を満たすノードに対応するレコードの集合と、第2の条件を満たすノードに対応するレコードの集合とを抽出するノード検索手段（3，4）と、前記入力手段から入力された第1の条件と第2の条件の間の祖孫関係を受け取り、前記ノード検索手段の抽出した一方のレコードの指す部分木索引の中に、他方のレコードの指すノードが存在するか否かを検索し、該当するノードが存在し、かつ前記祖孫関係を満足する場合に、該当ノードを第2の条件を満たすノードとして出力する接続関係検索手段（4，5）と、該接続関係検索手段の出力する該当ノードの持つ情報を前記データ記憶手段から抽出して表示する表示処理手段（6，7）とを備えることを特徴とする。

【0032】更に、本発明のデータ検索装置（請求項2）は、上記の構成に加え、前記部分木索引記憶手段は、更に、前記部分木索引のポインタの指す各ノードが、該部分木索引の根ノードとなるノードの子であるか子以外の子孫であるかを区別するデータを含むことを特徴とする。

【0033】また、本発明のデータ検索装置（請求項3）は、ノード間の関係が木あるいは木の集合で表現されるノードの集合から、第1の条件を満たすノードに対して祖孫関係を持ちかつ第2の条件を満たすノードを検索するデータ検索装置であって、第1の条件，第2の条件およびその間の祖孫関係を入力する入力手段（1，2）と、ノード，ノード間の関係およびノードの属性値を格納したデータ記憶手段（10）と、前記データ記憶手段に格納された各ノードごとに設けられ、当該ノードへのノードポインタと当該ノードの親ノードへのノードポインタを保持するレコードを指す親ポインタとから成るレコードの集合を保持するレコード集合保持手段（14）と、前記データ記憶手段に格納されたノードの属性値と該属性値を持つノードに対応する前記レコードへのポインタから成る索引を保持する索引保持手段（12）と、前記入力手段から入力された第1の条件および第2

の条件を受け取り、前記索引保持手段の索引から第1の条件を満たすノードに対応するレコードの集合と、第2の条件を満たすノードに対応するレコードの集合とを抽出するノード検索手段（3，4）と、前記入力手段から入力された第1の条件と第2の条件の間の祖孫関係を受け取り、前記ノード検索手段の抽出した一方のレコードの内、子孫である一方のレコードの親ポインタを根ノードまで辿り、辿った経路上のレコードのノードポインタの集合と、第2の条件を満たすレコードのノードポインタの集合との積集合を計算して得られるノードポインタの指すノードを該当ノードとして出力する接続関係検索手段（4，5）と、該接続関係検索手段の出力する該当ノードの持つ情報を前記データ記憶手段から抽出して表示する表示処理手段（6，7）とを備えることを特徴とする。

【0034】

【作用】本発明のデータ検索装置（請求項1）においては、ノード間の関係が木あるいは木の集合で表現されるノードの集合から、第1の条件を満たすノードに対して祖孫関係を持ち、かつ第2の条件を満たす第2のノードを検索する場合、まず、入力手段（1，2）が、第1の条件，第2の条件およびその間の祖孫関係を入力する。データ記憶手段（10）は、ノード，ノード間の関係およびノードの属性値を格納している。部分木索引記憶手段（9）は、ここでのデータ記憶手段（10）に格納された葉ノード以外の各ノードごとに設けられており、当該ノードを根ノードとする部分木を構成するノードへのポインタから成る部分木索引を保持する。

【0035】また、レコード集合保持手段（14）は、データ記憶手段（10）に格納された各ノードごとに設けられたレコードの集合を保持する。各レコードは、対応するノードへのポインタと前記部分木索引へのポインタとから構成される。索引保持手段（12）は、前記データ記憶手段に格納されたノードの属性値と該属性値を持つノードに対応する前記レコードへのポインタから成る索引を保持する。

【0036】そして、ノード検索手段（3，4）が、入力手段から入力された第1の条件および第2の条件を受け取り、前記索引保持手段の索引から第1の条件を満たすノードに対応するレコードの集合と、第2の条件を満たすノードに対応するレコードの集合とを抽出すると、接続関係検索手段（5，4）が、入力手段から入力された第1の条件と第2の条件の間の祖孫関係を受け取り、前記ノード検索手段の抽出した一方のレコードの指す部分木索引の中に、他方のレコードの指すノードが存在するか否かを検索する。該当するノードが存在し、かつ前記祖孫関係を満足する場合に、該当ノードを第2の条件を満たすノードとして出力する。出力された結果により、表示処理手段（6，7）は、該接続関係検索手段の出力する該当ノードの持つ情報を前記データ記憶手段から抽出して表示する。ここでは、部分木索引中にノードの属性値を保持しなくてもよく、索引の記憶容量を低減できる。

【0037】また、本発明のデータ検索装置（請求項2）においては、前記部分木索引記憶手段に保持される部分木索引のポインタの指す各ノードが、該部分木索引の根ノードとなるノードの子であるか子以外の子孫であるかを区別するデータを含む。これにより、ノードの検索では、子であるか、または子以外の子孫下であるかを区別した検索を行うことができる。

【0038】また、本発明のデータ検索装置（請求項3）においては、ノード間の関係が木あるいは木の集合で表現されるノードの集合から、第1の条件を満たすノードに対して祖孫関係を持ちかつ第2の条件を満たす第2のノードを検索する場合、入力手段（1，2）が、第1の条件，第2の条件およびその間の祖孫関係を入力する。データ記憶手段（10）は、ノード，ノード間の関係およびノードの属性値を格納しており、レコード集合保持手段（14）は、該データ記憶手段に格納された各ノードごとに設けられたレコードの集合を保持している。ここでのレコードは、当該ノードへのノードポインタと当該ノードの親ノードへのノードポインタを保持するレコードを指す親ポインタとから構成されている。

【0039】索引保持手段（12）は、前記データ記憶手段に格納された各ノードの属性値と該属性値を持つノードに対応する前記レコードへのポインタから成る索引を保持しているので、ノード検索手段（3，4）が、入力手段から入力された第1の条件および第2の条件を受け取り、前記索引保持手段の索引から第1の条件を満たすノードに対応するレコードの集合と第2の条件を満たすノードに対応するレコードの集合とを抽出する。また、ここでの接続関係検索手段（5，4）は、前記入力手段から入力された第1の条件と第2の条件の間の祖孫関係を受け取り、前記ノード検索手段の抽出した一方のレコードの内で、子孫である一方のレコードの親ポインタを根ノードまで辿り、辿った経路上のレコードのノードポインタの集合と、第2の条件を満たすレコードのノードポインタの集合との積集合を計算して得られるノードポインタの指すノードを該当ノードとして出力する。ここで出力された結果により、表示処理手段（6，7）が、接続関係検索手段の出力する該当ノードの持つ情報を前記データ記憶手段から抽出して表示する。

【0040】このように、本発明のデータ検索装置（請求項1）においては、第1の条件を満たすノードまたは第2の条件を満たすノードの一方が、他方を根ノードとする部分木に存在するかどうかを照合する部分木索引を設けて、第1の条件および第2の条件を満たすノードの検索を行う。このため、図9に示したような従来の転置ファイルではノードの属性値のデータを含むが、この部分木索引は、索引中にノードの属性値のデータを含む必

要がなくなり、それにより、索引の記憶容量を低減できることになる。

【0041】また、本発明のデータ検索装置（請求項3）においては、ノードへのポインタとそのノードを根とする部分木の索引へのポインタの組からなるレコードに、そのレコードに対応するノードの親ノードに対応するレコードへのポインタを持たせている。これにより、あるノードの親あるいは先祖ノードを検索する場合に、レコードの集合上での走査だけで済むようになり、先祖ノードの検索時間コストを小さくできる。また、1つのレコードの大きさは予じめ設定して決められており、かつ十分小さいので、図11および図12に示すような従来のデータの管理方法よりは、読み込み処理を速くできる。

【0042】

【実施例】以下、本発明の一実施例を図面を参照して具体的に説明する。図1は、本発明の一実施例にかかるデータ検索装置の要部の要成を示すブロック図である。図1において、1はキーボードなどの入力部、2は条件解析部、3はノード検索部、4はレコード記憶部、5は接続関係検索部、6は表示処理部、7はディスプレイ装置などの表示部、8はノード索引ファイル、9は部分木索引ファィル、10はデータファイルである。

【0043】ここでのノード索引ファイル8は、与えられた条件を満たすノードを検索するための索引を保持しており、部分木索引ファイル9は、各ノードを根とする部分木に対する索引を保持する。データファイル10は木構造で表現されるデータの集合を保持している。

【0044】入力部1が、検索のための条件（第1の条件，第2の条件，および親子関係あるいは祖孫関係の条件）を指定すると、条件解析部2が、入力部1によって指定された条件を解析する。ノード検索部3は、条件解析部2によって解析された条件に基づき、ノード索引ファイル8を読み込み、該当する条件を満たすノードの検索を行う。そして、レコード記憶部4において、ノード検索部3および接続関係検索部5によって検索されたレコードの集合を記憶する。また、接続関係検索部5は、部分木索引ファイル9から部分木索引を読み込んで、レコード記憶部4に保持されたレコード集合から親子関係あるいは祖孫関係を検索する。表示処理部6は、レコード記憶部4に保持された検索結果を基に、該当するノードをデータファイル10から読み込んで、表示部7に表示させる表示処理を行う。この結果、表示部7において検索された結果が表示される。

【0045】次に、ノード索引ファイル8，部分木索引ファイル9，およびデータファイル10におけるデータ構造の詳細を説明する。図2は、ノード索引ファイル，部分木索引ファイル，およびデータファイルにおけるデータ構造とその関係を説明する図である。図2に示すように、ノード索引ファイル8は、索引12およびレコード集合14から構成される。索引12はレコード集合14中で、与えられた条件を満たすノードへのポインタを持つレコードを検索するための索引となっており、この索引12により、一つのノードについて複数の条件を指定して検索することができる。

【0046】レコード集合14を構成している個々のレコード13は、親レコードへのポインタ15と、部分木索引へのポインタ16と、ノードへのポインタ17とから構成されており、親レコード，部分木索引，およびノードの間を関係付けている。このレコード13におけるポインタの指すノードが親ノードを持たない場合、つまり、該当のノードが根ノードの場合には、親レコードへのポインタ15は空ポインタとなっている。また、同様に、このレコード13におけるポインタの指すノードが葉ノードの場合、部分木索引へのポインタ16は空ポインタとなっている。

【0047】部分木索引ファイル9においては、各ノードを根とする部分木に対する部分木索引11を保持している。部分木索引11は、レコード13の中のノードへのポインタ17が指すノードを根とする部分木に対する索引となっており、接続関係18とノードへのポインタ19とが組になって構成されている。この部分木索引11により、あるレコード中のノードへのポインタをキーとして、接続関係18とノードへのポインタ19の組から接続関係18を検索する。接続関係18を表わすデータにおいては、該当するレコードに対するノードの接続関係を、記号コードのデータにより、子を“C”とし、子でない子孫を“D”として区別して保持している。

【0048】また、データファイル10は、木構造で表現されるデータの集合であるノードデータ21の集合を保持している。このノードデータ21は、ノードとしての木構造のリンク状態の示すポインタおよびその属性となる本体部のデータから構成されている（図11および図12を参照）。

【0049】次に、このようなデータ構造を有するノード索引ファイル8，部分木索引ファイル9，およびデータファイル10におけるレコードのデータを用いて、第1の条件を満たすノードの子あるいは子孫である第2の条件を満たすノードを検索する処理について説明する。図3は、第1の条件を満たすノードの子あるいは子孫である第2の条件を満たすノードを検索するデータ検索処理の一例を示すフローチャートである。

【0050】図1，図2，および図3を参照して、データ検索処理を説明する。処理を開始すると、まず、ステップ201において、入力部1からノードに対する第1の条件，第2の条件，および接続関係の条件の各々の条件を読み込む。次に、ステップ202において、入力部1によって読み込まれたノードに対する第1の条件，第2の条件，および接続関係の条件を、条件解析部2により解析する。次に、ステップ203において、ノード検

索部３はノード索引ファイル８から索引１２を読み込み、ステップ２０４において、ノード検索部３が、条件解析部２により解析された条件から、第１の条件を満たすノードを、索引１２を用いて検索する。その結果、得られたレコードの集合をレコード記憶部４に保持する。

【００５１】次に、ステップ２０５において、同じく、ノード検索部３は、条件解析部２によって解析された条件から、第２の条件を満たすノードを、索引１２を用いて検索し、その結果、得られたレコードの集合をレコード記憶部４に保持する。そして、次のステップ２０６において、子孫側のノードへのポインタを先祖側のノードの部分木索引の中で照合する。すなわち、接続関係検索部５が、前のステップ２０５の処理により、レコード記憶部４に保持された第２の条件を満たすレコード集合の中のレコードについて、そのレコード中のノードへのポインタを、前のステップ２０４の処理によりレコード記憶部４に保持された第１の条件を満たすレコード集合の中のレコードが指す部分木索引中で検索する。

【００５２】そして、ステップ２０７において、該当するノードが見つかったか否かを判定する。ノードが見つからなければ、ステップ２１１に進み、全てのレコードでの検索が終了したか否かを判定し、未処理のノードが残っている場合に、次のノードに対する処理に進む。また、この判定で、該当するノードが見つかった場合、次のステップ２０８に進み、更に、当該ノードにおいて接続関係が満たされているか否かを判定する。この判定で、接続関係が満たされていない場合に、前のステップ２０７と同様に、ステップ２１１に進み、全てのレコードでの検索が終了したか否かを判定し、未処理のノードが残っている場合に、次のノードに対する処理に進む。つまり、接続関係検索部５は、ノードへのポインタが部分木索引の中で見つかり、更に、指定された接続関係を満たしていることを検査する。

【００５３】これらの判定により、ノードへのポインタが部分木索引の中で見つかり、更に指定された接続関係を満たしていると、次に、ステップ２０９に進んで、該当するノードの情報をデータファイルから読み込み、次のステップ２１０において、ノードの内容情報を表示部に表示する。そして、ステップ２１１に進み、全てのレコードでの検索が終了した否かを判定し、未処理のノードが残っている場合には、ステップ２０６に戻り、ステップ２０６からの処理により、次のノードに対する処理を行う。一方、ステップ２１１の判定において、全てのレコードでの検索が終了していることが判定できると、一連の処理を終了する。

【００５４】このように、ここでのデータ検索処理では、第１の条件を満たすノードおよび第２の条件を満たすノードに対して、その一方が、他方を根とする部分木に存在するかどうかを、部分木索引により照合し、第１の条件および第２の条件を満たすノードの検索を行う。部分木索引は、索引中にノードの属性値のデータを含む必要がなくなり、それにより、索引のための記憶容量を低減して、データ検索ができるようになる。

【００５５】ところで、本実施例のデータ検索装置においては、図２に示すように、レコード集合１４における個々のレコード１３には、ノードへのポインタ１７とそのノードを根とする部分木索引へのポインタ１６との組に対し、更に、この各々のレコード１３に対応して、当該レコードに対応するノードの親ノードに対応するレコードへのポインタ１５を有している。これにより、あるノードの親あるいは先祖ノードを検索する場合には、レコード集合上で走査だけで済むようになっている。

【００５６】次に、このような各々のレコード１３の中の親レコードへのポインタ１５を用いた場合のデータ検索処理について説明する。図４は、第１の条件を満たすノードの親あるいは先祖である第２の条件を満たすノードを検索するデータ検索処理を、各々のレコード１３の中の親レコードへのポインタ１５を用いて行う場合の処理手順を示すフローチャートである。

【００５７】次に、前述の場合と同様に、図１，図２，および図４を参照して、データ検索をレコード集合上の走査のみで行う場合のデータ検索処理について説明する。処理を開始すると、ステップ３０１において、入力部１からノードに対する第１の条件，第２の条件，および接続関係の条件の各々の条件を読み込む。次に、ステップ３０２において、入力部１によって読み込まれたノードに対する第１の条件，第２の条件，および接続関係の条件を、条件解析部２により解析する。次に、ステップ３０３において、ノード検索部３はノード索引ファイル８から索引１２を読み込み、ステップ３０４において、ノード検索手段３が、条件解析部２により解析された条件から、第１の条件を満たすノードを、索引１２を用いて検索する。その結果、得られたレコードの集合をレコード記憶部４に保持する。

【００５８】次に、ステップ３０５において、同じく、ノード検索部３は、条件解析部２によって解析された条件から、第２の条件を満たすノードを、索引１２を用いて検索し、その結果、得られたレコードの集合をレコード記憶部４に保持する。そして、次のステップ３０６において、子孫側のノードへのポインタを持つレコードから、親レコードのポインタを辿る。すなわち、ノード検索部３は、前のステップ３０４の処理により、レコード記憶部４に保持された第１の条件を満たすレコード集合中のひとつのレコードについて、その親レコードへのポインタを辿り、得られた親レコードをレコード記憶部４に保持する。

【００５９】そして、次のステップ３０７において、指定された接続関係は親子関係であるか否かを判定する。つまり、条件解析部２によって解析された接続関係の条件を参照して、親子関係かまたは祖孫関係かを判断す

15

る。この判定の結果、親子関係であれば、それで良いので、ステップ３１０に進み、全てのレコードでの検索が終了した否かを判定し、未処理のノードが残っている場合に、次のノードに対するレコードの処理に進む。

【００６０】また、このステップ３０７の判定で、親子関係でない場合、つまり、指定された接続関係が祖孫関係である場合、ステップ３０８に進み、ルートノードに辿り着いたか否かを判定し、ルートノードに辿り着いていない場合、更に、ステップ３０９において、親ノードへのポインタを辿り、得られた親レコードをレコード記憶部４に保持し、続いて、ステップ３０８に戻り、ルートノードに辿り着いたか否かを判定する処理を繰り返す。ルートノードに辿り着いたことが判定できると、それで良いので、ステップ３１０に進み、全てのレコードでの検索が終了した否かを判定し、未処理のノードが残っている場合に、次のノードに対するレコードの処理に進む。

【００６１】すなわち、ノード検索部３により、レコードのポインタにより辿られた親レコードがルートレコード（ルートノードへのポインタを持つレコード）かどうか判断し、辿られた親レコードがルートレコードでないとき、ノード検索部３は親レコードの親レコードへのポインタを辿り、得られた親レコードをレコード記憶部４に保持する操作を繰り返し行う。この結果、レコード記憶部４には順次に辿られた親レコードが保持される。

【００６２】そして、指定された接続関係が親子関係の場合に、または親レコードがルートレコードである場合に、次のステップ３１０において、レコード記憶部４に保持された第１の条件を満たすレコード集合中の全てのレコードについて処理を終了したか否かを判定する。この判定で、未処理のレコードが残っていると判定される場合には、ステップ３０６に戻って、残りのレコードに対する処理を同様に繰り返し行う。また、全てのレコードについて処理を終了したことが確認できると、次に、ステップ３１１において、レコード記憶部４に保持された第２の条件を満たすレコードのノードへのポインタの集合と、親（先祖）レコードの持つノードへのポインタの集合との積集合を計算する。この集合の計算結果は、条件として与えられた各々の条件を満たしていることになるので、次に、ステップ３１２において、該当レコード内のノードへのポインタを辿って、条件に該当するノード情報をデータファイルから読み込み、次に、ステップ３１３において、読み込まれたノードの情報を表示部７に表示して、一連の処理を終了する。

【００６３】次に、このようなデータ検索処理を木構造を有する検索対象データに対して行う場合の具体例について説明する。図５は、検索対象データの一例を示す説明図である。ここでは、木構造を有する検索対象データとしては、木構造の論理構造を有する文書データを例にして説明する。図５に示すように、検索対象データの文

16

書データは、有向順序木によって表現されており、文書名の「報告書」を根ノードとして、各ノードは、例えば、章，節，段落などのタイプを有している。各ノードは他の属性を持っているが、図５においては、ノードのタイプのみを示している。

【００６４】図５に示すような検索対象データに対して、図３に示したようなデータ検索処理の手順により、「図を含む章」を条件として検索する場合、次のようにして、ここでのデータ検索処理が行なわれる。図６は、図３に示す処理フローによりデータ検索処理を行う場合の各々のレコードの集合のデータの参照関係を示す図である。図６を参照して説明を続けると、図６においては、右側に検索対象データのうちの関連部分のみ示しており、ハッチングしたブロックのノード６１が条件を満たすノードとなっている。この場合に、それぞれ参照されるレコードの集合を左側に示している。

【００６５】この場合のデータ検索処理において、「図を含む章」という条件は、ノードに関する第１の条件が「タイプが章である」という条件であり、ノードに関する第２の条件は「タイプが図である」という条件である。また、接続関係の条件は「祖孫関係」という条件である。

【００６６】このようなデータ検索処理では、まず、ノード検索部３によって、ノードに関する第１の条件の「タイプが章である」という条件を満たすノードへのポインタを持つレコード群（６０１～６０４）と、同じくノードに関する第２の条件となっている「タイプが図である」という条件を満たすノードへのポインタを持つレコード群（６０５，６０６）とが得られる。

【００６７】次に、接続関係検索部５によって、第２の条件を満たすレコード６０５におけるノードへのポインタ６０７と、同じく第２の条件を満たすレコード６０６におけるノードへのポインタ６０８とにより指示されているノードの中に対して、更に、第１の条件を満たすレコード群（６０１～６０４）における部分木索引の中での探索を行う。

【００６８】この場合には、レコード群（６０１～６０４）の部分木索引ポインタ（６１０～６１３）が指示している部分木索引６２において、接続関係６２ａを参照し、更に、ノードへのポインタ６２ｂを参照して、共通に指示されているノードを探索する。その結果、ここでは、部分木索引ポインタ６１３が指示する部分木索引６２の中で、ポインタ６０９が指示するノードが、前述のレコード６０６のポインタ６０８が指示するノードと一致する。したがって、条件に適合する求められるべきノードは、レコード６０４中のノードへのポインタ６１４が指すノードとなる。

【００６９】次に、図４に示したようなデータ検索処理の手順により、同様にして「表を含む章」を条件として検索する場合について、その具体的なレコードのデータ

参照について説明する。この場合にも、前述の場合と同様に検索対象データは、図５に示したような木構造を有する文書データとする。

【００７０】図７は、図４に示す処理フローによりデータ検索処理を行う場合の各々のレコードの集合のデータ参照関係を説明する図である。この場合のデータ検索の条件としては「表を含む章」を条件として、該当するノードのデータ検索する。図７に基づいて説明する。図７においては、右側に検索対象データのうちの関連部分のみ示しており、ここで、ハッチングしたブロックのノード（７１，７２）が、条件を満たすノードとなっている。そして、この場合にそれぞれに参照されるレコードの集合を左側に示している。

【００７１】この場合のデータ検索処理において、「表を含む章」という条件は、第１の条件が「タイプが章である」という条件であり、また、第２の条件が「タイプが表である」という条件となっている。接続関係の条件は「祖孫関係」という条件である。

【００７２】したがって、このようなデータ検索処理では、まず、ノード検索部３により、ノードに関する第１の条件の「タイプが章である」という条件を満たすノードへのポインタを持つレコード群（７０１～７０４）と、同じくノードに関する第２の条件となっている「タイプが表である」という条件を満たすノードへのポインタを持つレコード群（７０６，７０７）とが得られる。次に、これらのレコード群の間のポインタを辿る操作を行い、レコード７０６およびレコード７０７の親レコードへのポインタを辿り、レコード７０１およびレコード７０５をそれぞれに得る。

【００７３】ここで指定された接続関係の条件は「祖孫関係」であり、また、得られたレコード７０１およびレコード７０５は、共にルートレコードではないので、更に、親レコードへのポインタを辿る操作を行い、次に、レコード７００およびレコード７０２をそれぞれに得る。レコード７００はルートレコードであるが、レコード７０２はルートレコードではないので、レコード７０２から更に親レコードへのポインタを辿り、最終的にレコード７００を得る。

【００７４】このポインタを辿る操作処理の結果、第１の条件の「タイプが章である」条件を満たすレコードから対応のノードへのポインタ（７０９～７１２）と、親レコードを順次辿って得られたレコード群（７００，７０１，７０２，７０５）の中の各々のノードへのポインタ（７０８，７０９，７１０，７１３）との積集合の計算を行う。この積集合の計算の結果、与えられた条件を満たすノードへのポインタとして、ポインタ７０９およびポインタ７１０が得られる。したがって、条件に適合する求められるべきノードは、ポインタ７０９およびポインタ７１０が指示するノード７１およびノード７２となる。

【００７５】

【発明の効果】以上説明したように、本発明のデータ検索装置によれば、木構造で表現されるデータの集合の中で、第１の条件を満たすノードと親子関係あるいは祖孫関係を持ち、かつ第２の条件を満たすノードを検索する場合、この検索の際に用いられる部分木索引では、ノードの属性値の情報を含む必要がなくなり、転置ファイルよりも小さくできるようになる。このため、索引の記憶容量を低減することができる。また、第１の条件を満たすノードの親あるいは先祖である第２の条件を満たすノードの検索の場合にも、例えば、レコードの集合の中で走査だけで済むので、２次記憶から主記憶へのデータの読み込み回数を減らすことができ、全体としてデータ検索の処理のスループットを向上させることができる。

【図面の簡単な説明】

【図１】　図１は本発明の一実施例にかかるデータ検索装置の要部の要成を示すブロック図、

【図２】　図２はノード索引ファイル，部分木索引ファイル，およびデータファイルにおけるデータ構造とその関係を説明する図、

【図３】　図３は第１の条件を満たすノードの子あるいは子孫である第２の条件を満たすノードを検索するデータ検索処理の一例を示すフローチャート、

【図４】　図４は第１の条件を満たすノードの親あるいは先祖である第２の条件を満たすノードを検索するデータ検索処理を各々のレコードの中の親レコードへのポインタ１５を用いて行う場合の処理手順を示すフローチャート、

【図５】　図５は検索対象データの一例を示す説明図、

【図６】　図６は図３に示す処理フローによりデータ検索処理を行う場合の各々のレコードの集合のデータの参照関係を示す図、

【図７】　図７は図４に示す処理フローによりデータ検索処理を行う場合の各々のレコードの集合のデータ参照関係を説明する図、

【図８】　図８は文書の論理構造を木構造で表現している一例を示す図、

【図９】　図９はノードの親子関係あるいは祖孫関係の条件を用いる検索処理を部分木に対する転置ファイルを用いて行う場合の検索方法を説明する図、

【図１０】　図１０は一つのノードからその先祖ノードを辿るために必要なパスを有するデータ構造の一例を示す図、

【図１１】　図１１はノードの情報を２次記憶上に連続に割り付ける第１の態様を示す図、

【図１２】　図１２はノードの情報を２次記憶上に分割して割り付ける第２の態様を示す図である。

【符号の説明】

１…入力部、２…条件解析部、３…ノード検索部、４…レコード記憶部、５…接続関係検索部、６…表示処理

19　　20

部、７…表示部、８…ノード索引ファイル、９…部分木索引ファイル、１０…データファイル、１１…部分木索引、１２…索引、１３…レコード、１４…レコード集合、１５…親レコードへのポインタ、１６…部分木索引へのポインタ、１７…ノードへのポインタ、１８…接続関係、１９…ノードへのポインタ、２１…ノードデータ、６１…検索されるべきノード、６２…部分木索引、６２ａ…接続関係、６２ｂ…ノードへのポインタ、７１，７２…検索されるべきノード、９０…根ノード、９１…Ａノード、９２…Ｂノード、９０ａ…根ノードの転置ファイル、９１ａ…Ａノードの転置ファイル、９２ａ…Ｂノードの転置ファイル、１１０…ノードの情報、１１１…親ノードへのポインタ、１１２…子ノードへのポインタ、１１３…ノードの属性、１２１…リンク情報、１２１ａ…親ノードへのポインタ、１２１ｂ…子ノード集合へポインタ、１２１ｃ…属性集合へのポインタ、１２２…子ノード集合、１２３…属性集合、６０１～６０６…レコード、６０７～６０９…ノードへのポインタ、６１０～６１３…部分木索引へのポインタ、６１４…ノードへのポインタ、７００～７０７…レコード、７０８～７１５…ノードへのポインタ。

【図１】

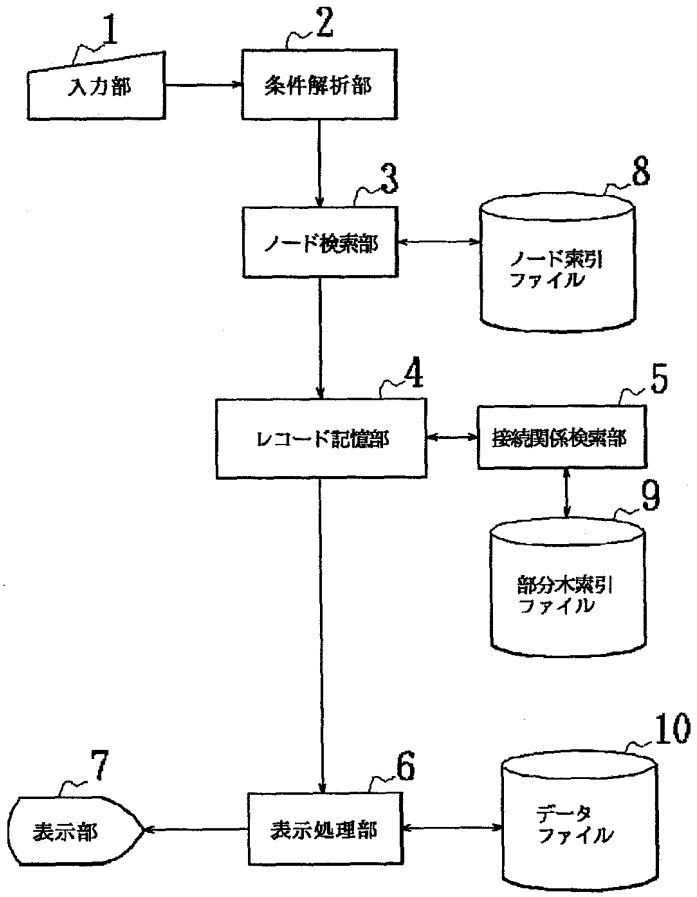

図１　データ検索装置の構成

【図５】

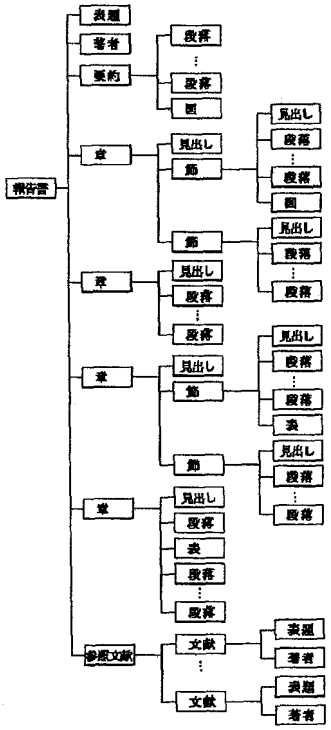

図５　検索対象データの一例

【図１１】

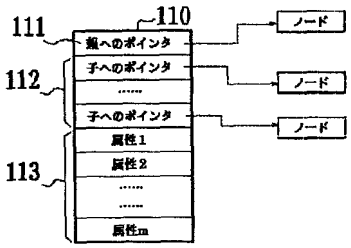

図１１　ノード情報の記憶領域割付け方法の例

【図２】

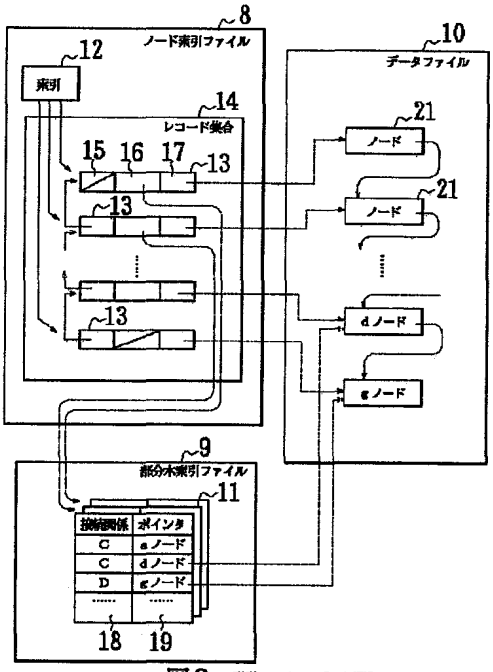

図２　索引ファイルのデータ構造

【図６】

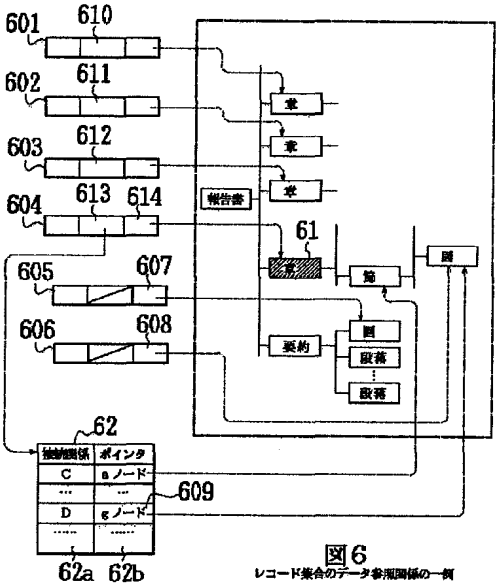

図６
レコード集合のデータ参照関係の一例

【図８】

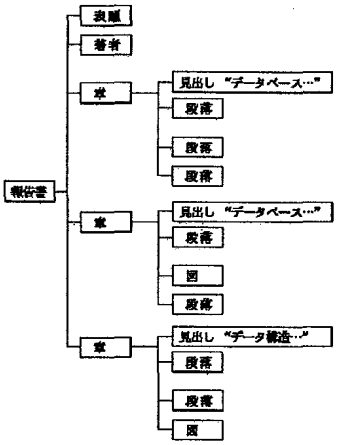

図８　文書の論理構造の一例

【図７】

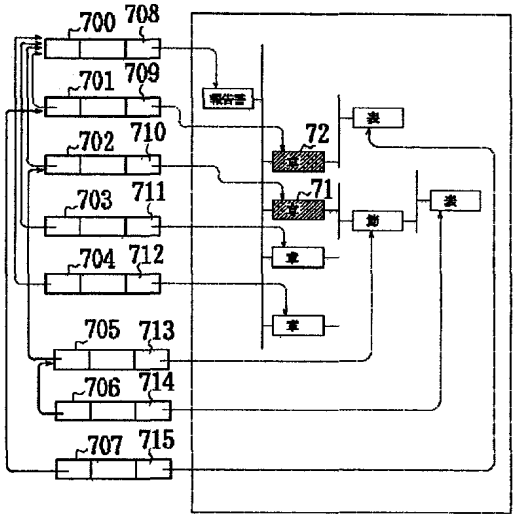

図７
レコード集合のデータ参照関係の他の一例

【図３】

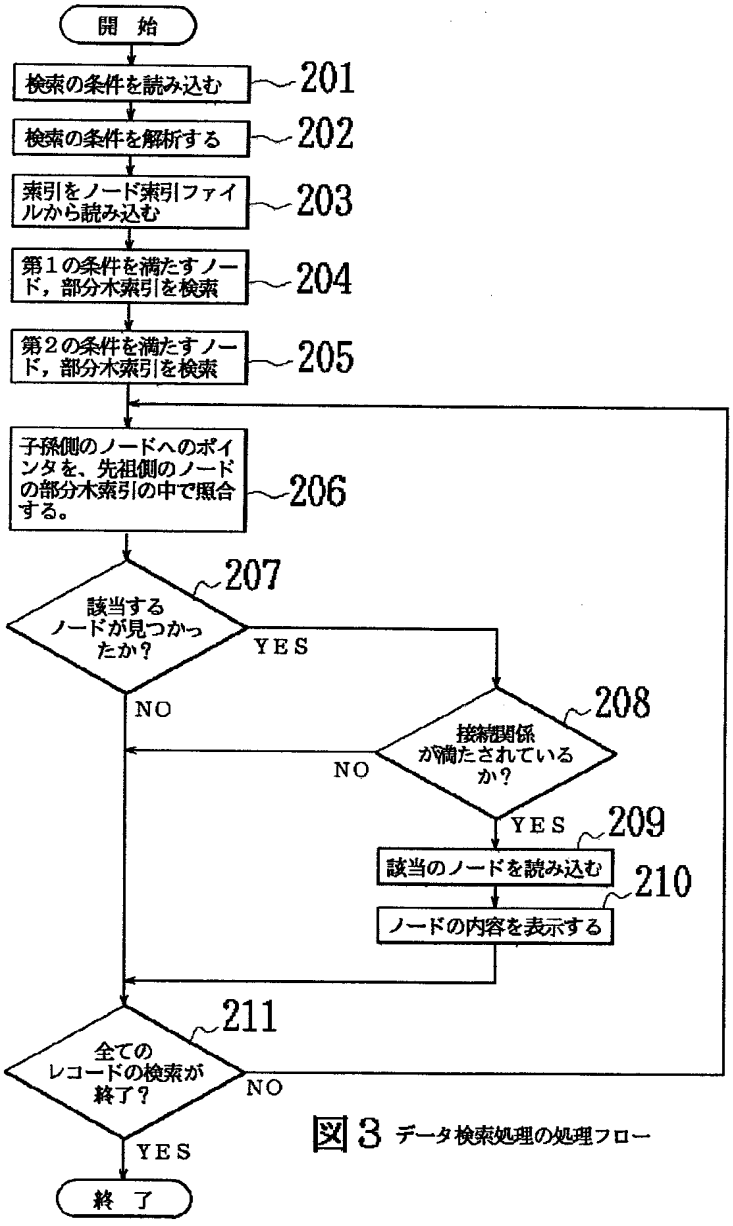

図３　データ検索処理の処理フロー

【図４】

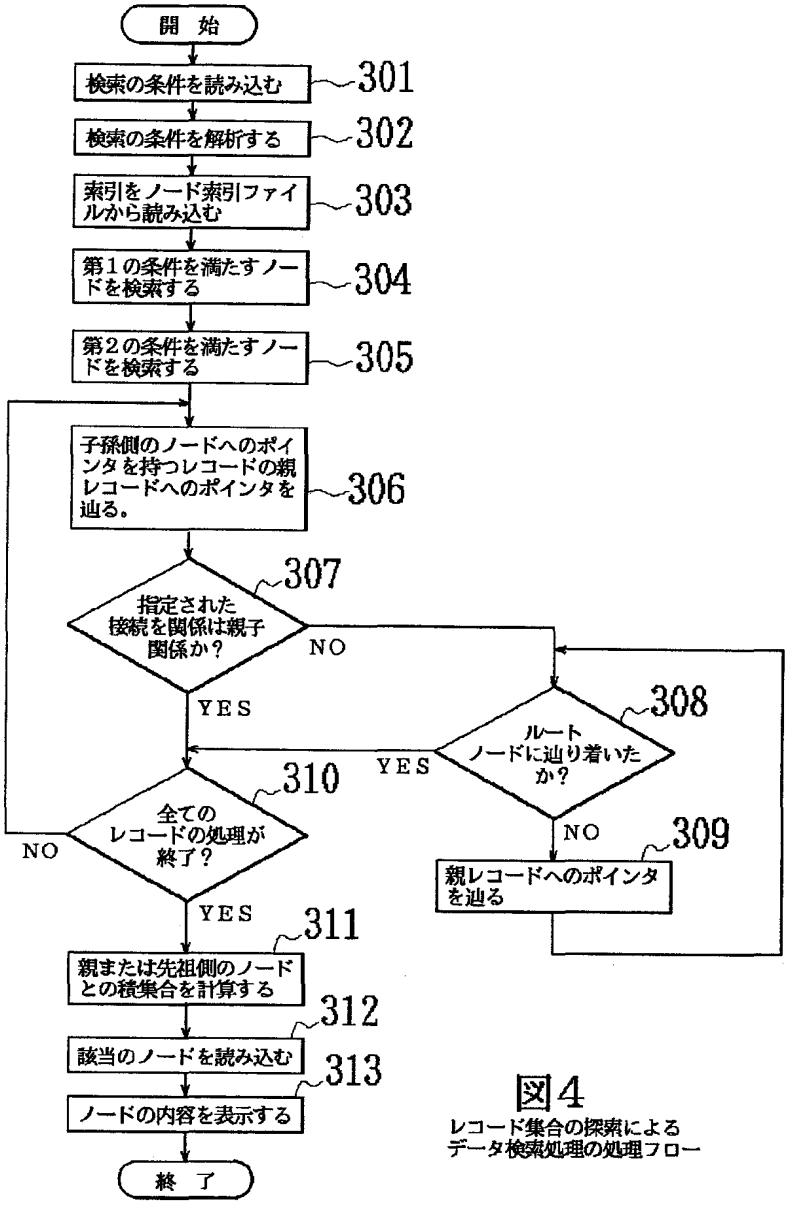

図４
レコード集合の探索による
データ検索処理の処理フロー

【図９】

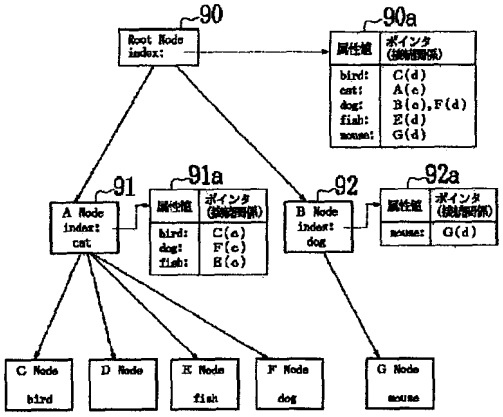

図９　参照ファイルによる検索

【図１０】

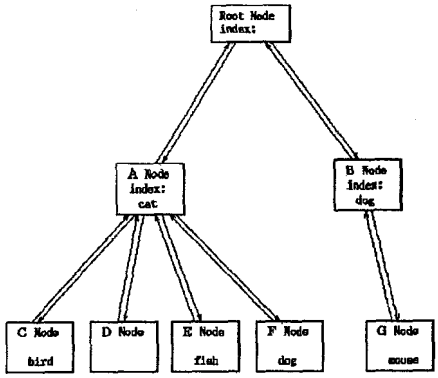

図１０　双方向リンクによる検索

【図１２】

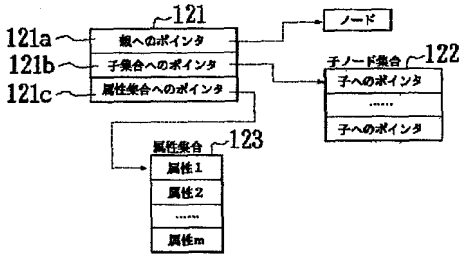

図１２　ノード情報の記憶領域割付け方法の他の例